UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/535,065	10/24/2005	Catherine Robert	S1022.81242US00	1848

46329 7590 07/16/2009

STMicroelectronics Inc.
c/o WOLF, GREENFIELD & SACKS, P.C.
600 Atlantic Avenue
BOSTON, MA 02210-2206

EXAMINER	
VICARY, KEITH E	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
07/16/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/535,065	ROBERT ET AL.	
	Examiner	Art Unit	
	Keith Vicary	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 May 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-15 are pending in this office action and presented for examination.

Claims 1, 7-8, and 11-15 are newly amended by amendment filed 5/13/2009.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-15 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nexus 5001 Forum: Standard for a Global Embedded Processor Debug Interface (Nexus 5001 Forum) in view of Argade et al. (Argade) (US 5724505).

4. **Consider claim 1**, Nexus 5001 Forum discloses a method for transmitting digital messages (page 52 of 150, section 6.2, transmission), on execution of an instruction sequence by a microprocessor (page 59 of 150, last paragraph, message is output by the target processor whenever there is a change of program flow), through output terminals of a monitoring circuit integrated on the microprocessor (Table 7-1, Auxiliary Pins required per interface), at least one digital message of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect

branch messages), the method comprising, the steps of: determining whether the jump is associated with a jump instruction explicitly indicating an address of a destination instruction of the jump (Table 6-6 and Table 6-7, which shows the resulting messages based on the determination; a determination is inherent for the appropriate message to be sent); after it is determined that that the address of the destination instruction is explicitly indicated in the jump instruction: assigning a first value to a first set of bits of at least one digital message to provide an explicit jump message, and transmitting the explicit jump message; and when it is determined that the address of the destination instruction is not explicitly indicated in the jump instruction: assigning a second value to the first set of bits of at least one digital message to provide an implicit jump message indicating an occurrence of an implicit jump, and transmitting the implicit jump message (Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3; when no, Table 6-7 shows that the TCODE set of bits will be equal to 4; page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose adding a field to the implicit jump message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps, wherein the field is added after it is determined that the address of the destination instruction is not explicitly indicated in the jump instructions.

On the other hand, Argade does disclose of identifying a jump as an implicit jump from among several types of implicit jumps by assigning to a set of bits a specific value

(col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of implicit jumps being type_1 and type_2, in col. 5, lines 49-65).

Argade's teaching of identifying an implicit jump from among several types of implicit jumps enables real-time program tracing which enables the user to re-construct a full program trace (Argrade, col. 5, lines 35-38).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Argade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Note that the combination of Argade and Nexus would teach the claimed limitations, which is explained as follows.

Argade teaches of identifying types of implicit jumps in trace information. The invention of Nexus entails sending out trace information via digital messages. It would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade into the digital messages of Nexus in order to decrease cost and space, for example (e.g. incorporating the type of implicit jump in the tracing mechanisms already existing in Nexus, as opposed to creating some entirely new method of trace information transmission solely for the use of conveying implicit jump type information, would obviate the cost of this hypothetical new method of trace information transmission as well as the space necessary to implement this hypothetical new method of trace information transmission. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of

adding a field to the digital message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps.

Additionally, it would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade, which identifies types of implicit jumps, in only the implicit jump messages of Nexus, in order to increase efficiency, as an explicit jump message of Nexus would logically have no use for information identifying a type of implicit jump (as an explicit jump is not an implicit jump at all), and forgoing the field in explicit jump messages would decrease the size of the explicit jump messages, thus increasing transmission efficiency. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the implicit jump message wherein the field is added only when the address of the destination instruction is not explicitly indicated in the jump instructions.

5. **Consider claim 7**, Nexus 5001 Forum discloses means for detection of a jump on execution of an instruction sequence by the microprocessor (Nexus 51001 Forum, Table 6-6 and 6-7); means for storing data characteristic of the detected jump (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages); means for generating at least one digital message based on the stored characteristic data, the at least one digital message comprising a first set of bits, wherein: the first set of bits is set to a first value when the jump is associated with a jump instruction of the instruction sequence explicitly indicating an address of a destination instruction of the jump to

provide an explicit jump message, and the first set of bits set to a second value when the jump is associated with a jump instruction of the instruction sequence not explicitly indicating the address of the destination instruction to provide an implicit jump message indicating an occurrence of an implicit jump (Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3; when no, Table 6-7 shows that the TCODE set of bits will be equal to 4); and means for transmitting the generated at least one digital message (page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose, after it is determined that the address of the destination instruction is not explicitly indicated in the jump instruction and the first set of bits is set to the second value, the generation means adds a field to the implicit jump message, the field comprising a second set of bits identifying a type of the implicit jump from among several implicit jump types.

On the other hand, Argade does disclose of identifying a jump as an implicit jump from among several types of implicit jumps by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of implicit jumps being type_1 and type_2, in col. 5, lines 49-65).

Argade's teaching of identifying an implicit jump from among several types of implicit jumps enables real-time program tracing which enables the user to re-construct a full program trace (Argrade, col. 5, lines 35-38).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Argade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Note

that the combination of Argade and Nexus would teach the claimed limitations, which is explained as follows.

Argade teaches of identifying types of implicit jumps in trace information. The invention of Nexus entails sending out trace information via digital messages. It would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade into the digital messages of Nexus in order to decrease cost and space, for example (e.g. incorporating the type of implicit jump in the tracing mechanisms already existing in Nexus, as opposed to creating some entirely new method of trace information transmission solely for the use of conveying implicit jump type information, would obviate the cost of this hypothetical new method of trace information transmission as well as the space necessary to implement this hypothetical new method of trace information transmission. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the digital message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps.

Additionally, it would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade, which identifies types of implicit jumps, in only the implicit jump messages of Nexus, in order to increase efficiency, as an explicit jump message of Nexus would logically have no use for information identifying a type of implicit jump (as an explicit jump is not an implicit jump at all), and forgoing the field in explicit jump messages would decrease the size of the explicit jump messages, thus increasing transmission efficiency. The implementation of

the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the implicit jump message wherein the field is added only when the first set of bits is set to the second value.

6. **Consider claim 8**, Nexus 5001 Forum discloses detecting a jump in the execution of the instruction sequence from an initial instruction to a jump destination instruction, wherein the jump destination instruction is different from an instruction following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages, the detection is inherent in the transmission of these messages); determining whether the jump is associated with a jump instruction explicitly indicating an address of the jump destination instruction (Table 6-6 and Table 6-7, the message which is generated is dependent upon this determination); generating at least one digital message upon the detection of the jump (Table 6-6 and 6-7), after it is determined that the jump is associated with a jump instruction not explicitly indicating the address of the jump destination instruction: determining that the jump is implicit, generating the at least one digital message as an implicit jump message indicating an occurrence of an implicit jump (Table 6-7 shows that the TCODE set of bits will be equal to 4) when the jump is not implicit, generating the at least one digital message as an explicit jump message (Table 6-6); and transmitting the at least one digital message (page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose adding an additional field to the implicit jump message, wherein the additional field includes a value identifying a type of the implicit jump.

On the other hand, Argade does disclose of identifying a jump as an implicit jump from among several types of implicit jumps by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of implicit jumps being type_1 and type_2, in col. 5, lines 49-65).

Argade's teaching of identifying an implicit jump from among several types of implicit jumps enables real-time program tracing which enables the user to re-construct a full program trace (Argrade, col. 5, lines 35-38).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Argade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Note that the combination of Argade and Nexus would teach the claimed limitations, which is explained as follows.

Argade teaches of identifying types of implicit jumps in trace information. The invention of Nexus entails sending out trace information via digital messages. It would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade into the digital messages of Nexus in order to decrease cost and space, for example (e.g. incorporating the type of implicit jump in the tracing mechanisms already existing in Nexus, as opposed to creating some entirely new method of trace information transmission solely for the use of conveying implicit

jump type information, would obviate the cost of this hypothetical new method of trace information transmission as well as the space necessary to implement this hypothetical new method of trace information transmission. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the digital message, the additional field including a value identifying a type of the implicit jump.

Additionally, it would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade, which identifies types of implicit jumps, in only the implicit jump messages of Nexus, in order to increase efficiency, as an explicit jump message of Nexus would logically have no use for information identifying a type of implicit jump (as an explicit jump is not an implicit jump at all), and forgoing the field in explicit jump messages would decrease the size of the explicit jump messages, thus increasing transmission efficiency. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the implicit jump message wherein the field is added only when the jump is implicit.

7. **Consider claim 11**, Nexus 5001 Forum discloses a monitoring circuit integrated on a microprocessor for (Table 7-1, Auxiliary Pins required per interface; also, some form of monitoring circuit is inherent given that trace messages are being sent based on program flow): detecting, on execution of an instruction sequence by the microprocessor, a jump from an initial instruction to a jump destination instruction,

wherein the jump destination instruction is different from an instruction following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages, the detection is inherent in the transmission of these messages); only when the jump is implicit, providing at least one digital message as an implicit jump message transmitted on the execution of the instruction sequence by the microprocessor and indicating an occurrence of an implicit jump (Table 6-7 shows that the TCODE set of bits will be equal to 4; page 52 of 150, section 6.2, transmission); when the jump is not implicit, providing the at least one digital message as an explicit jump message (see above citation); an analysis tool to reconstitute the instruction sequence based on the at least one digital message; and at least one monitoring terminal to provide the at least one digital message from the monitoring circuit to the analysis tool (Table 7-1, Auxiliary Pins required per interface; also, some form of monitoring circuit is inherent given that trace messages are being sent based on program flow; page 5 of 150, last paragraph, program trace visibility, development tools, page 51 of 150, first paragraph, the tool; page 58 of 150, section 6.4.4, program trace).

However, Nexus 5001 Forum does not explicitly disclose only after it is determined that the jump is associated with a jump instruction not explicitly indicating an address of the jump destination instruction, adding a field to the implicit jump message, wherein the field includes a value identifying a type of the implicit jump.

On the other hand, Argade does disclose of identifying a jump as an implicit jump from among several types of implicit jumps by assigning to a set of bits a specific value

(col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of implicit jumps being type_1 and type_2, in col. 5, lines 49-65).

Argade's teaching of identifying an implicit jump from among several types of implicit jumps enables real-time program tracing which enables the user to re-construct a full program trace (Argrade, col. 5, lines 35-38).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Argade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Note that the combination of Argade and Nexus would teach the claimed limitations, which is explained as follows.

Argade teaches of identifying types of implicit jumps in trace information. The invention of Nexus entails sending out trace information via digital messages. It would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade into the digital messages of Nexus in order to decrease cost and space, for example (e.g. incorporating the type of implicit jump in the tracing mechanisms already existing in Nexus, as opposed to creating some entirely new method of trace information transmission solely for the use of conveying implicit jump type information, would obviate the cost of this hypothetical new method of trace information transmission as well as the space necessary to implement this hypothetical new method of trace information transmission. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of

adding a field to the digital message, wherein the field includes a value identifying a type of the implicit jump.

Additionally, it would have been further obvious to one of ordinary skill in the art at the time of the invention to include the trace information of Argade, which identifies types of implicit jumps, in only the implicit jump messages of Nexus, in order to increase efficiency, as an explicit jump message of Nexus would logically have no use for information identifying a type of implicit jump (as an explicit jump is not an implicit jump at all), and forgoing the field in explicit jump messages would decrease the size of the explicit jump messages, thus increasing transmission efficiency. The implementation of the trace information of Argade into the digital messages of Nexus thus teaches the limitations of adding a field to the implicit jump message, wherein the field includes a value identifying a type of the implicit jump.

8. **Consider claim 2**, Nexus 5001 Forum discloses a step of assigning to a third set of bits of the at least one digital message a value corresponding to a number of instructions executed by the microprocessor since a last executed instruction of the instruction sequence for which a digital message associated with a jump was transmitted (page 59, Table 6-6 and 6-7, the I-CNT field).

9. **Consider claim 3**, Nexus 5001 Forum discloses a step of assigning to a fourth set of bits of the implicit jump message a value representative of the address of the destination instruction (Nexus 5001 Forum, Table 6-7, U-ADDR).

10. **Consider claim 4**, Argade discloses the type of the implicit jump corresponds to a jump resulting from a jump instruction of the instruction sequence containing a reference of a register that stores data representative of the destination instruction address (col. 5, lines 59-65, register indirect jump or call).

11. **Consider claim 5**, Argade discloses a jump type corresponds to a jump forced by the microprocessor, the destination instruction corresponding to an instruction comprising a series of specific instructions which are different from instructions of the instruction sequence (col. 5, lines 49-52, hardware interrupt).

12. **Consider claim 6**, Argade discloses the type of the implicit jump corresponds to a jump forced by the microprocessor, the destination instruction being an instruction of the instruction sequence (col. 5, lines 59-65, register indirect jump or call).

13. **Consider claim 9**, Nexus 5001 Forum and Argade disclose detecting the jump further comprises determining whether the jump is associated with a jump instruction of the instruction sequence explicitly indicating an address of a jump destination instruction of the jump instruction (Nexus 5001 Forum, page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages; it is inherent a determination takes places as either of the two messages are formed as a result of whether the address is explicitly indicated or not); and generating at least one digital message upon the detection of the jump comprises: when it is determined that the jump instruction explicitly indicates the

address of the jump destination instruction, assigning a first value to a first set of bits of the at least one digital message to provide the explicit jump message (Nexus 5001 Forum, Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3); and when it is determined that the jump instruction does not explicitly indicate the address of the jump destination instruction: assigning a second value to the first set of bits of the at least one digital message to provide the implicit jump message (Nexus 5001 Forum, when no, Table 6-7 shows that the TCODE set of bits will be equal to 4); and assigning to the additional field of the implicit jump message comprising a second set of bits a third value identifying the type of the implicit jump (Argade, as explained in the independent claim, col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2 in col. 5, lines 49-65).

14. **Consider claim 10**, Nexus 5001 Forum and Argade disclose the at least one digital message is transmitted through output terminals of a monitoring circuit integrated on the microprocessor (Nexus 5001 Forum, Table 7-1, Auxiliary Pins required per interface; Argade, col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal, col. 4, 59-65, JTAG interface and port, col. 4, lines 51, 59-65, HDS block)

15. **Consider claims 12-15**, Nexus 5001 Forum and Argade disclose a modification of the at least one digital message to indicate the type of the implicit jump comprises adding the field only when the at least one digital message is provided as the implicit jump message (see the rejections of the independent claim which further explain how this limitation is met). With regard to claim 14, Nexus 5001 Forum discloses the digital message is generated in accordance with a standard (i.e. Nexus 5001 Forum Standard for a Global Embedded Processor Debug Interface).

Response to Arguments

16. Applicant's arguments filed 5/13/2009 have been fully considered but they are not persuasive.

17. In the first two paragraphs of page 8, applicant argues both that Nexus does not teach a certain set of claimed limitations, and that Argade likewise does not teach a certain set of claimed limitations.

However, while it may be true that Nexus *by itself* and Argade *by itself* does not teach the aforementioned claimed limitations, the examiner has been consistently taking the position that *the modification of Nexus by Argade* does teach the aforementioned claimed limitations. This position is explained both in the rejection and response to arguments in the previous and current office actions.

18. In the paragraph spanning page 8 and 9, Applicant argues with corresponding citations that Argade clearly uses the INSTR_TYPE signal to determine whether to

record or discard address information, and accordingly does not teach the aforementioned claimed limitations.

It is again first noted that the examiner has relied upon Nexus as modified by Argade to teach the claimed limitations, and not Argade in isolation. Furthermore, Argade's use of using a signal which indicates whether a jump is implicit or explicit to determine whether to record or discard address information does not run contrary to Nexus: Nexus's message only includes the U-ADDR information when the jump is indirect/implicit as opposed to direct/explicit. Given that the type of message which is generated is dependent on whether the jump is implicit or explicit, it is readily seen that Nexus too uses some information/signal which indicates whether a jump is implicit or explicit to determine whether to record or discard address information as well.

19. Applicant argues in the paragraph spanning page 9 and 10 that for proper operation of Argade, all three types of the INSTR_TYPE signals are required to be sent. Applicant further notes that the TBC block of Argade determines if the INSTR_TYPE is type_3.

However, examiner is uncertain what applicant is conveying with this argument. Examiner's rejection involves implementing Argade's general recognition that it is desirable to identify an implicit jump among several types of implicit jumps into the invention and overall environment of Nexus. Examiner is not trying to modify the invention of Argade, nor implement Argade's entire invention into the environment of

Nexus. Therefore, the details of Argade's *specific implementation* which require three types of INSTR_TYPE signals to be sent and Argade's specific TBC block is irrelevant.

Applicant further argues in the first indented paragraph of page 10 that, essentially, Argade's specific implementation does not teach the claimed limitations. However, examiner again notes that the rejection relies upon Nexus as modified by Argade. It would also be readily recognized that an obviousness rejection using two arts does not necessarily require that every single element of a secondary art be implemented into the primary art.

20. In the paragraph spanning pages 10 and 11, Applicant further argues that Argade does not teach the claimed limitation, and provides numerous citations showing how exactly the invention of Argade functions.

However, examiner again notes that the rejection relies upon Nexus as modified by Argade. Examiner recommends that applicant instead argue why the invention of Nexus, as modified by the teaching of Argade and as explained in depth in the rejection above and previous response to arguments, as opposed to arguing why Argade does not teach the claimed limitations.

21. Applicant argues on the first indented paragraph of page 11 that there is no suggestion in Argade that it may be appropriate to use only the type_1 and type_2 types of discontinuities and not the type_3.

However, as previously explained, the invention of Nexus *already distinguishes between implicit and explicit jumps* given its teaching of using an indirect branch message and a direct branch message. It would be unreasonable to take the position that the benefits of identifying an implicit jump from among several types of implicit jumps are only applicable in the context of the specific method that Argade uses to convey branch information and not in any other method of conveying branch information.

22. Applicant argues in the second indented paragraph of page 11 that none of the references teaches or suggests "adding a field to the implicit jump message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps."

However, as cited and explained above, Nexus teaches of the use of implicit jump messages which contain fields comprising bits which identify the destination address of the corresponding implicit jump. Argade teaches that information which identifies a type of implicit jump amongst multiple types of implicit jumps is desirable. It would have been very readily recognized to one of ordinary skill in the art at the time of the invention that, if one were to modify the above teaching of Nexus with the above teaching of Argade, the overall combination would entail adding a field to the implicit jump message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps.

As a simple analogy, consider two banks, Bank A and Bank B. Bank A snail-mails interest statements of a certain layout to a customer, and Bank B e-mails interest statements of another layout to a customer. Say Bank A decided to include in its interest statements additional information regarding the projected account balance at the end of the year. It would be very readily recognized that Bank B could likewise include in its interest statements that same additional information *without necessitating a change in the delivery method or the layout*. Analogously, it would be very readily recognized to one of ordinary skill in the art at the time of the invention that Nexus could incorporate Argade's identification of a type of implicit jump in *Nexus's own* method and format of communication (i.e. messages with fields) without necessitating that Nexus also adopt the method and format of communication of Argade.

23. Applicant further argues in the second indented paragraph of page 11 that none of the references suggests that "adding a field to the implicit jump message, the field comprising a second set of bits identifying a type of the implicit jump from among several types of implicit jumps" would increase efficiency.

However, examiner cited this motivation in the context of why one of ordinary skill in the art at the time of the invention would *only* add this field to the implicit jump message and not the explicit jump message. It would be very readily recognized to one of ordinary skill in the art at the time of the invention that arbitrarily adding bits that aren't used to a message is inefficient (e.g. a waste of bandwidth). This concept is already embodied in Nexus: the direct branch message does not include a U-ADDR

field because it is not needed. Similarly, a direct branch message would not need a field which indicates a type of indirect jump, because the message is directed toward a *direct* jump.

Applicant argues that the type_3 is required in Argade. However, this is irrelevant as examiner's rejection does not involve removing the type_3 from Argade.

24. Applicant argues in the third indented paragraph of page 11 that it is not clear why one would not provide the type_3 signal of Argade for the direct branch message of Nexus.

However, in Argade, the type_3 signal is used to indicate a direct branch. Nexus' environment already entails the use of a direct branch message, with an associated TCODE, to indicate a direct branch. It is unclear as to why Applicant *would* provide an *additional* indicator in Nexus' direct branch message to redundantly indicate that the direct branch message correlates to a direct message.

25. The above response to arguments are analogously applicable to the other independent claims as argued from pages 12-13.

Conclusion

26. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

27. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Keith Vicary whose telephone number is (571)270-1314. The examiner can normally be reached on Monday - Thursday, 6:15 a.m. - 5:45 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on 571-272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Eddie P Chan/
Supervisory Patent Examiner, Art Unit 2183

/Keith Vicary/
Examiner, Art Unit 2183